JP11328155

Publication Title:

Symbolic calculation system, symbolic calculation method and parallel circuit simulation system

Abstract:

The coefficient matrix, corresponding to the simultaneous linear equations to be solved, is divided into a plurality of row sets. The row sets as divided are processed in a parallel fashion, and entries specifying the nonzero elements contained in the first to n<th >row sets are added to the entry sets E1 to En. Moreover, in regard to each row set, fill-ins which take place at the time of eliminating the i<th >variable are obtained in a parallel fashion, and entries specifying the fill-ins are added to the entry sets E1 to En. The coefficient matrix is compressed using those entry sets E1 to En.

---

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平11－328155

(43)公開日　平成11年(1999)11月30日

(51)Int.Cl.[6]	識別記号	Ｆ Ｉ		
Ｇ０６Ｆ　17/12		Ｇ０６Ｆ　15/324		
17/50		15/60	６６２Ｇ	

審査請求　有　請求項の数５　ＯＬ　（全 14 頁）

(21)出願番号　特願平10－127738

(22)出願日　平成10年(1998) 5月11日

(71)出願人　000004237
日本電気株式会社
東京都港区芝五丁目７番１号

(72)発明者　蜂屋　孝太郎
東京都港区芝五丁目７番１号　日本電気株式会社内

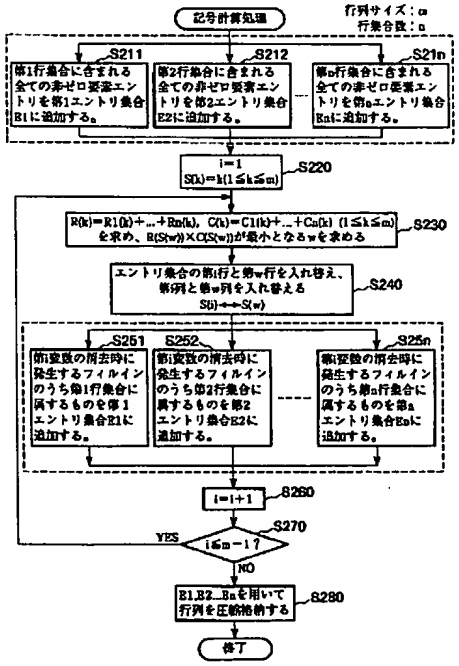
(74)代理人　弁理士　古澗　聡　（外１名）

(54)【発明の名称】　記号計算システム及び方法、並びに並列回路シミュレーションシステム

(57)【要約】

【課題】　線形連立方程式の求解における記号計算処理で、扱える係数行列の規模を大きくし、また、計算時間を短縮する。

【解決手段】　求解対象となる線形連立方程式に対応する係数行列を、少なくとも１行ずつからなる複数の行集合に分割する。分割された行集合のそれぞれは、並列に処理されて、第１～第ｎ行集合に含まれている値が「０」でない要素のエントリが、それぞれに対応するエントリ集合Ｅ１～Ｅｎに追加される（ステップＳ２１１～Ｓ２１ｎ）。また、第ｉ変数を消去するときに発生するフィルインを行集合毎に並列して求め、そのフィルインに対応するエントリをエントリ集合Ｅ１～Ｅｎに追加する（ステップＳ２５１～Ｓ２５ｎ）。そして、これらエントリ集合Ｅ１～Ｅｎのすべてを用いて、係数行列を圧縮する（ステップＳ２８０）。



1

【特許請求の範囲】

【請求項１】線形連立方程式の求解のための記号計算を行う記号計算システムであって、

求解の対象となる線形連立方程式を示す行列を、それぞれ少なくとも前記行列の１行分からなる複数の行集合に分割する行列分割手段と、

前記行列分割手段によって分割された複数の行集合のそれぞれに対応し、対応する行集合に含まれる値が０でない要素のエントリを、それぞれに対応するエントリ集合に追加する複数の第１のエントリ追加手段と、

値が１から前記行列の行数まで順次変化される第１の変数が示す変数の消去時に必要となる演算回数が最小となる第２の変数を順次求めるピボット選択手段と、

前記行列中の前記第１の変数が示す行と前記第２の変数が示す行、及び前記行列中の前記第１の変数が示す列と前記第２の変数が示す列とをそれぞれ交換するピボット交換手段と、

前記複数の第１のエントリ追加手段のそれぞれに対応し、前記ピボット交換手段によって行及び列が交換された行列について変数の消去時に発生するフィルインを対応する行集合毎に求め、該フィルインのエントリを対応する第１のエントリ追加手段でそれぞれエントリが追加されたエントリ集合にさらに追加する複数の第２のエントリ追加手段と、

前記第１、第２のエントリ追加手段によってエントリが追加された複数のエントリ集合を用いて、前記行列を圧縮する行列圧縮手段と、を備えることを特徴とする記号計算システム。

【請求項２】前記複数の第１、第２のエントリ追加手段は、対応するもの同士で同一の手段によって構成され、前記複数の第１、第２のエントリ追加手段はそれぞれ、対応する行集合の行内エントリの数を記憶する行内エントリ数配列を管理する手段と、列内エントリの数を記憶する列内エントリ数配列を管理する手段とを有し、

前記ピボット選択手段は、前記行内エントリ数配列が記憶する行内エントリ数をそれぞれ配列の添字が等しいもの同士で加算する手段と、前記列内エントリ数配列が記憶する列内エントリ数をそれぞれ配列の添字が等しいもの同士で加算する手段と、これらの手段によってそれぞれ加算された行内エントリ数と列内エントリ数とを添字が等しいもの同士で乗算する手段と、これらの乗算結果中の値が最も小さいものの添字を選択する選択手段とを有し、前記選択手段によって選択された添字を前記第２の変数とすることを特徴とする請求項１に記載の記号計算システム。

【請求項３】前記ピボット交換手段は、行列中の交換された行及び列を示す置換配列を記憶する手段と、該手段に記憶された置換配列を前記複数の第２のエントリ追加手段のそれぞれに供給する手段とを備え、

前記複数の第２のエントリ追加手段は、前記ピボット交換手段から供給された置換配列に従って、行及び列が交換された行列について変数の消去時に発生するフィルインを対応する行集合毎に求めることを特徴とする請求項１または２に記載の記号計算システム。

2

【請求項４】線形連立方程式の求解のための記号計算を行う記号計算方法であって、

求解の対象となる線形連立方程式を示す行列を、それぞれ少なくとも前記行列の１行分からなる複数の行集合に分割する行列分割ステップと、

前記行列分割ステップで分割された複数の行集合のそれぞれに応じて並列して動作し、対応する行集合に含まれる値が０でない要素のエントリを、それぞれに対応するエントリ集合に追加する第１のエントリ追加ステップと、

値が１から前記行列の行数まで順次変化される第１の変数が示す変数の消去時に必要となる演算回数が最小となる第２の変数を順次求めるピボット選択ステップと、

前記行列中の前記第１の変数が示す行と前記第２の変数が示す行、及び前記行列中の前記第１の変数が示す列と前記第２の変数が示す列とをそれぞれ交換するピボット交換ステップと、

前記第１のエントリ追加ステップで並列に行われるそれぞれの処理に対応して並列に動作し、前記ピボット交換ステップで行及び列が交換された行列について変数の消去時に発生するフィルインを対応する行集合毎に求め、該フィルインのエントリを第１のエントリ追加ステップでそれぞれエントリが追加されたエントリ集合にさらに追加する第２のエントリ追加ステップと、

前記第１、第２のエントリ追加ステップでエントリが追加された複数のエントリ集合を用いて、前記係数行列を圧縮する行列圧縮ステップと、を備えることを特徴とする記号計算方法。

【請求項５】シミュレーションの対象となる回路を複数の部分回路に分割する部分回路分割手段と、

前記部分回路分割手段で分割された複数の部分回路のそれぞれに対応し、対応する部分回路の回路方程式に対応する部分回路行列を求める複数の部分回路行列計算手段と、

前記複数の部分回路行列計算手段によって計算された部分回路行列のそれぞれに記号計算を行う複数の部分回路記号計算手段と、

前記複数の記号計算手段によって記号計算されたそれぞれの結果にさらに数値計算処理を行い、対応する部分回路の回路方程式の解を求める複数の部分回路数値計算手段と、

前記複数の部分回路数値計算手段によって求められた各部分回路の回路方程式の解に従って、部分回路を結合した結合回路の回路方程式に対応する結合回路行列を求める結合回路行列計算手段と、

前記結合回路行列計算手段によって計算された結合回路

行列に記号計算を行う結合回路記号計算手段と、
前記結合回路記号計算手段によって記号計算された結果に更に数値計算処置を行い、結合回路の回路方程式の解を求める結合回路数値計算手段と、を備え、
前記結合回路数値計算手段は、
前記結合回路行列計算手段によって求められた結合回路行列を、それぞれ少なくとも前記行列の１行分からなる複数の行集合に分割する行列分割手段と、
前記行列分割手段によって分割された複数の行集合のそれぞれに対応し、対応する行集合に含まれる値が０でない要素のエントリを、それぞれに対応するエントリ集合に追加する複数の第１のエントリ追加手段と、
値が１から前記結合回路行列の行数まで順次変化される第１の変数が示す変数の消去時に必要となる演算回数が最小となる第２の変数を順次求めるピボット選択手段と、
前記結合回路行列中の前記第１の変数が示す行と前記第２の変数が示す行、及び前記結合回路行列中の前記第１の変数が示す列と前記第２の変数が示す列とをそれぞれ交換するピボット交換手段と、
前記複数の第１のエントリ追加手段のそれぞれに対応し、前記ピボット交換手段によって行及び列が交換された結合回路行列行列について変数の消去時に発生するフィルインを対応する行集合毎に求め、該フィルインのエントリを対応する第１のエントリ追加手段でそれぞれエントリが追加されたエントリ集合にさらに追加する複数の第２のエントリ追加手段と、
前記第１、第２のエントリ追加手段によってエントリが追加された複数のエントリ集合を用いて、前記結合回路行列を圧縮する行列圧縮手段と、を備えることを特徴とする並列回路シミュレーションシステム。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、線形連立方程式の解を求めるための記号計算を行う記号計算システム及び方法、並びに線形連立方程式の解を求めることにより回路のシミュレーションを行う並列回路シミュレーションシステムに関する。

【０００２】

【従来の技術】線形連立方程式の解を求めるための処理は、一般に、記号計算処理と数値計算処理とからなる。この線形連立方程式の求解処理における記号計算処理の一例が、"Foudamentals of Computer-Aided Circuit Simulation" (William J. McCalla著、１９８８年刊) の第３章 "Sparse Matrix Methods" に記載されている。

【０００３】図９は、上記従来例の線形連立方程式の求解処理における記号計算処理を示すフローチャートである。まず、求解の対象となる線形連立方程式を表す係数行列全体に対して１つのエントリ集合Ｅを用意し、係数行列中の値が「０」でない要素のエントリをエントリ集合Ｅに順次追加していく。ここで、第ｋ行に含まれるエントリ数及び第ｋ列に含まれるエントリ数を示す配列Ｒ（ｋ）、Ｃ（ｋ）も係数行列全体に対して１つずつ用意して、エントリ登録時に値を更新していく（ステップＳ４１０）。

【０００４】次に、何番目の変数を消去するかを示す変数ｉに「１」を代入し、初期化する（ステップＳ４２０）。次に、Ｒ（ｗ）×Ｃ（ｗ）が最小となるｗの値を求める。ここで、Ｒ（ｗ）、Ｃ（ｗ）の値を求めるときに、既に消去された変数と式に対応する行列部分のエントリや、対角要素となるエントリ数はカウントされない（ステップＳ４３０）。

【０００５】ｗの値が求められると、エントリ集合Ｅの第ｉ行と第ｗ行とを交換し、また、第ｉ列と第ｗ列とを交換する（ステップＳ４４０）。次に、第ｉ変数の消去時に発生するフィルインを求め、エントリ集合Ｅに追加する（ステップＳ４５０）。

【０００６】ステップＳ４５０の処理が終了すると、変数ｉの値を「１」だけインクリメントされ（ステップＳ４６０）、さらに変数ｉの値がｍ－１（ｍ：係数行列のサイズ）以下であるかどうかが判定される（ステップＳ４７０）。ステップＳ４７０で変数ｉの値がｍ－１以下であると判定されているときは、ステップＳ４３０の処理に戻る。このような処理の繰り返しによって、最適なピボット順序とフィルインの発生位置が求められる。

【０００７】一方、ステップＳ４７０で変数ｉの値がｍの値に等しくなったと判定されると、ステップＳ４１０及びステップＳ４５０でエントリが追加されたエントリ集合Ｅを用いて行列を圧縮し、圧縮した行列データを格納する（ステップＳ４８０）。そして、このフローチャートの処理を終了する。そして、数値計算処理に進むこととなる。

【０００８】

【発明が解決しようとする課題】しかしながら、上記従来例には、次のような問題点があった。まず第１に、記号計算処理を分散メモリ型の並列コンピュータで実行する場合に、記号計算処理を単一のノード上で逐次実行しなければならなかった。このため、ノード内の局所メモリの容量によって、扱える係数行列のサイズが制限されてしまうという問題点があった。

【０００９】また、第２に、係数行列のサイズ（行及び列のそれぞれの数（行数及び列数は一致する））をｍとした場合の記号計算処理の時間は、ｍの１乗から３乗程度のオーダで増加するため、係数行列のサイズが大きくなり、エントリ集合に追加するエントリ数が膨大になると、記号計算処理を実用的な時間内で終了することができなくなってしまうという問題点があった。

【００１０】このような問題点は、線形連立方程式の解を求めることによって回路のシミュレーションを行う場合にも生じていた。

【００１１】本発明は、上記従来例の問題点を解消するためになされたものであり、扱える行列の規模が大きく、計算時間が短い記号計算システム及び方法を提供することを目的とする。

【００１２】本発明は、また、大規模な回路のシミュレーションを高速に行うことができる並列回路シミュレーションシステムを提供することを目的とする。

【００１３】

【課題を解決するための手段】上記目的を達成するため、本発明の第１の実施の形態にかかる記号計算システムは、線形連立方程式の求解のための記号計算を行う記号計算システムであって、求解の対象となる線形連立方程式を示す行列を、それぞれ少なくとも前記行列の１行分からなる複数の行集合に分割する行列分割手段と、前記行列分割手段によって分割された複数の行集合のそれぞれに対応し、対応する行集合に含まれる値が０でない要素のエントリを、それぞれに対応するエントリ集合に追加する複数の第１のエントリ追加手段と、値が１から前記行列の行数まで順次変化される第１の変数が示す変数の消去時に必要となる演算回数が最小となる第２の変数を順次求めるピボット選択手段と、前記行列中の前記第１の変数が示す行と前記第２の変数が示す行、及び前記行列中の前記第１の変数が示す列と前記第２の変数が示す列とをそれぞれ交換するピボット交換手段と、前記複数の第１のエントリ追加手段のそれぞれに対応し、前記ピボット交換手段によって行及び列が交換された行列について変数の消去時に発生するフィルインを対応する行集合毎に求め、該フィルインのエントリを対応する第１のエントリ追加手段でそれぞれエントリが追加されたエントリ集合にさらに追加する複数の第２のエントリ追加手段と、前記第１、第２のエントリ追加手段によってエントリが追加された複数のエントリ集合を用いて、前記行列を圧縮する行列圧縮手段と、を備えることを特徴とする。

【００１４】上記計算システムによれば、エントリ集合へのエントリの追加を複数の第１、第２のエントリ追加手段によって並列して行うことができる。このため、線形連立方程式の解を求める場合の記号計算を高速に行うことができる。

【００１５】また、上記計算システムを分散メモリ型の並列コンピュータで実現する場合には、複数の第１、第２のエントリ追加手段を実現するためのノードが有する局所メモリに記憶すべきデータ量が、１つのノードでエントリ集合へのエントリ追加を行った場合に比べて、ほぼ行列の分割数の逆数をかけたものとすることができる。このため、各ノードが有する局所メモリの容量によって、扱える行列のサイズが制限されることが少なくなり、大規模な線形連立方程式の求解処理も行えるようになる。

【００１６】上記記号計算システムにおいて、前記複数の第１、第２のエントリ追加手段は、対応するもの同士で同一の手段、例えば、並列コンピュータの同一のノードによって構成されたものとすることができる。この場合、前記複数の第１、第２のエントリ追加手段はそれぞれ、対応する行集合の行内エントリの数を記憶する行内エントリ数配列を管理する手段と、列内エントリの数を記憶する列内エントリ数配列を管理する手段とを有するものとでき、前記ピボット選択手段は、前記行内エントリ数配列が記憶する行内エントリ数をそれぞれ配列の添字が等しいもの同士で加算する手段と、前記列内エントリ数配列が記憶する列内エントリ数をそれぞれ配列の添字が等しいもの同士で加算する手段と、これらの手段によってそれぞれ加算された行内エントリ数と列内エントリ数とを添字が等しいもの同士で乗算する手段と、これらの乗算結果中の値が最も小さいものの添字を選択する選択手段とを有し、前記選択手段によって選択された添字を前記第２の変数とすることができる。

【００１７】また、上記記号計算システムにおいて、前記ピボット交換手段は、行列中の交換された行及び列を示す置換配列を記憶する手段と、該手段に記憶された置換配列を前記複数の第２のエントリ追加手段のそれぞれに供給する手段とを備えるものとしてもよい。前記複数の第２のエントリ追加手段は、前記ピボット交換手段から供給された置換配列に従って、行及び列が交換された行列について変数の消去時に発生するフィルインを対応する行集合毎に求めるものとすることができる。

【００１８】このような置換配列を用いることによって、行列或いは行集合のやり取りをしなくても、第２のエントリ追加手段がフィルインを求めることができる。

【００１９】上記目的を達成するため、本発明の第２の観点にかかる記号計算方法は、線形連立方程式の求解のための記号計算を行う記号計算方法であって、求解の対象となる線形連立方程式を示す行列を、それぞれ少なくとも前記行列の１行分からなる複数の行集合に分割する行列分割ステップと、前記行列分割ステップで分割された複数の行集合のそれぞれに応じて並列して動作し、対応する行集合に含まれる値が０でない要素のエントリを、それぞれに対応するエントリ集合に追加する第１のエントリ追加ステップと、値が１から前記行列の行数まで順次変化される第１の変数が示す変数の消去時に必要となる演算回数が最小となる第２の変数を順次求めるピボット選択ステップと、前記行列中の前記第１の変数が示す行と前記第２の変数が示す行、及び前記行列中の前記第１の変数が示す列と前記第２の変数が示す列とをそれぞれ交換するピボット交換ステップと、前記第１のエントリ追加ステップで並列に行われるそれぞれの処理に対応して並列に動作し、前記ピボット交換ステップで行及び列が交換された行列について変数の消去時に発生するフィルインを対応する行集合毎に求め、該フィルインのエントリを第１のエントリ追加ステップでそれぞれエ

ントリが追加されたエントリ集合にさらに追加する第２のエントリ追加ステップと、前記第１、第２のエントリ追加ステップでエントリが追加された複数のエントリ集合を用いて、前記係数行列を圧縮する行列圧縮ステップと、を備えることを特徴とする。

【００２０】上記目的を達成するため、本発明の第３の観点にかかる並列回路シミュレーションシステムは、シミュレーションの対象となる回路を複数の部分回路に分割する部分回路分割手段と、前記部分回路分割手段で分割された複数の部分回路のそれぞれに対応し、対応する部分回路の回路方程式に対応する部分回路行列を求める複数の部分回路行列計算手段と、前記複数の部分回路行列計算手段によって計算された部分回路行列のそれぞれに記号計算を行う複数の部分回路記号計算手段と、前記複数の記号計算手段によって記号計算されたそれぞれの結果にさらに数値計算処理を行い、対応する部分回路の回路方程式の解を求める複数の部分回路数値計算手段と、前記複数の部分回路数値計算手段によって求められた各部分回路の回路方程式の解に従って、部分回路を結合した結合回路の回路方程式に対応する結合回路行列を求める結合回路行列計算手段と、前記結合回路行列計算手段によって計算された結合回路行列に記号計算を行う結合回路記号計算手段と、前記結合回路記号計算手段によって記号計算された結果に更に数値計算処置を行い、結合回路の回路方程式の解を求める結合回路数値計算手段と、を備え、前記結合回路数値計算手段は、前記結合回路行列計算手段によって求められた結合回路行列を、それぞれ少なくとも前記行列の１行分からなる複数の行集合に分割する行列分割手段と、前記行列分割手段によって分割された複数の行集合のそれぞれに対応し、対応する行集合に含まれる値が０でない要素のエントリを、それぞれに対応するエントリ集合に追加する複数の第１のエントリ追加手段と、値が１から前記結合回路行列の行数まで順次変化される第１の変数が示す変数の消去時に必要となる演算回数が最小となる第２の変数を順次求めるピボット選択手段と、前記結合回路行列中の前記第１の変数が示す行と前記第２の変数が示す行、及び前記結合回路行列中の前記第１の変数が示す列と前記第２の変数が示す列とをそれぞれ交換するピボット交換手段と、前記複数の第１のエントリ追加手段のそれぞれに対応し、前記ピボット交換手段によって行及び列が交換された結合回路行列行列について変数の消去時に発生するフィルインを対応する行集合毎に求め、該フィルインのエントリを対応する第１のエントリ追加手段でそれぞれエントリが追加されたエントリ集合にさらに追加する複数の第２のエントリ追加手段と、前記第１、第２のエントリ追加手段によってエントリが追加された複数のエントリ集合を用いて、前記結合回路行列を圧縮する行列圧縮手段と、を備えることを特徴とする。

【００２１】【発明の実施の形態】以下、添付図面を参照して、本発明の実施の形態について説明する。

【００２２】［第１の実施の形態］図１は、この実施の形態に適用される並列コンピュータの構成を示すブロック図である。

【００２３】図示するように、この並列コンピュータ１は、相互結合網１０を介して結合された複数のノード１１～１１から構成されている。ノード１１～１１は、それぞれ要素プロセッサ１１ａ～１１ａと、局所メモリ１１ｂ～１１ｂを備える。要素プロセッサ１１ａ～１１ａと局所メモリ１１ｂ～１１ｂとは、それぞれバス１１ｃ～１１ｃを介して接続されており、また、バス１１ｃ～１１ｃは、相互結合網１０に接続されている。

【００２４】要素プロセッサ１１ａ～１１ａは、相互結合網１０を介して互いにメッセージを交換し合う。また、要素プロセッサ１１ａ～１１ａは、相互結合網１０を介して他のノードが有する局所メモリ１１ｂ～１１ｂに遠隔アクセスする。これらによって、並列コンピュータ１は、あるノードから他の１つ以上のノードへ処理を依頼し、依頼した処理の並列処理を可能としている。

【００２５】局所メモリ１１ｂ～１１ｂはそれぞれ、対応する要素プロセッサ１１ａ～１１ａが実行するプログラムを記憶すると共に、要素プロセッサ１１ａ～１１ａのワークエリアとして使用される。

【００２６】並列コンピュータ１の相互結合網１０には、入力装置２１及び外部記憶装置２２を有するワークステーション２が接続されている。ワークステーション２の入力装置２１は、計算対象となる線形連立方程式（或いはその係数行列）を入力する。また、ワークステーション１は、係数行列を並列コンピュータ１に渡し、線形連立方程式の求解処理を依頼する。

【００２７】以下、この実施の形態において、図１の並列コンピュータで実行される線形連立方程式の求解処理について、説明する。図２は、この実施の形態にかかる線形連立方程式の求解処理の概略を示すフローチャートである。

【００２８】線形連立方程式の求解処理は、エントリ登録処理（ステップＳ１１０）と、対角ピボッティング処理（ステップＳ１２０）と、行列の圧縮格納処理（ステップＳ１３０）と、ＬＵ分割処理（ステップＳ１４０）と、前進代入・後進代入処理（ステップＳ１５０）とを順次実行することによって行われる。これらのうちで、ステップＳ１１０～Ｓ１３０までが記号計算処理であり、ステップＳ１４０とＳ１５０とが数値計算処理である。

【００２９】ステップＳ１１０のエントリ登録処理では、ワークステーション２から渡された係数行列のうちの値が「０」でない要素の位置がエントリ集合Ｅに追加される。この処理は、後述する行集合毎に並列して行われるものであり、初めに行集合毎のエントリ集合Ｅを空

集合とし、係数行列中の値が「０」でない要素の行ｉと列ｊとのエントリ(ｉ，ｊ)をそれぞれのエントリ集合Ｅに追加するという処理を全ての要素に対して行うものである。

【００３０】ステップＳ１２０の対角ピボッティング処理では、ステップＳ１４０のＬＵ分解処理での演算回数ができるだけ少なくなるように、係数行列の行交換と列交換とが行われる。この処理には、元の係数行列では値が「０」であった要素でＬＵ分解処理の過程で値が「０」でなくなる要素（フィルイン）のエントリを行集合毎にエントリ集合Ｅに追加する処理が含まれている。

【００３１】ステップＳ１３０の行列の圧縮格納処理では、ステップＳ１２０までの処理で得られたエントリ集合Ｅを用いて、係数行列及びＬＵ分解後に得られる行列の値が「０」でない要素の値のみを格納するための領域が確保され、係数行列の要素の値が格納される。

【００３２】ステップＳ１４０のＬＵ分割処理では、係数行列を左下三角行列Ｌと右上三角行列Ｕとの積に因数分解する処理が行われる。ここでは、係数行列が格納された領域に上書きをしていくことによって処理が進められる。

【００３３】ステップＳ１５０の前進代入・後進代入処理では、係数行列で表される線形連立方程式の解が求められる。この処理では、例えば、係数行列で表される線形連立方程式がＡｘ＝ｂ（Ａ：係数行列、ｘ：解ベクトル、ｂ：右辺ベクトル）としたときに、Ｌｙ＝ｂを解いてｙの解を求める前進代入処理と、Ｕｘ＝ｙを解いてｘの解を求める後進代入処理とが行われる。

【００３４】図３は、図２の記号計算処理（ステップＳ１１０～Ｓ１３０）を詳細に示すフローチャートである。このフローチャートにおけるステップＳ２１１～Ｓ２１１が図２のステップＳ１１０に対応し、ステップＳ２２０～Ｓ２７０が図２のステップＳ１２０に対応し、ステップＳ２８０が図２のステップＳ１３０に対応する。

【００３５】ユーザは、ワークステーション２の入力装置２１を操作して、解を求める対象となる線形連立方程式を入力する。そして、入力装置２１からワークステーション２に入力した線形連立方程式の解を求めるための指示を入力する。すると、ワークステーション２は、線形連立方程式の係数行列（行列サイズをｍとする）を求め、さらに複数の行集合（集合数をｎ（ｎ≦ｌ）とする）に分ける。そして、ワークステーション２は、この係数行列を並列コンピュータ１に渡すと共に、記号計算処理の依頼をする。

【００３６】並列コンピュータ１のノード１１～１１の要素プロセッサ１１ａ～１１ａうち、記号計算処理の依頼を受け取ったノード（ノード１１とする）の要素プロセッサ１１ａは、渡された係数行列を行集合毎にノード１１～１１中のｎ個のノードに相互結合網１０を介して送信する（但し、１つは自ノード１１内でプロセス間通信によって送信してもよい）。

【００３７】係数行列中の行集合を受け取ったｎ個のノード（ノード１１～１ｎとする）は、それぞれが並列に動作する。これらのノード１１～１ｎの要素プロセッサ１１ａ～１ｎａは、それぞれ行集合毎に要素を空集合とするエントリ集合Ｅ１～Ｅｎを用意し、自ノード内の局所メモリ１１ｂ～１ｎｂに領域を確保する。また、第ｋ行に含まれるエントリ数及び第ｋ列に含まれるエントリ数を示す配列Ｒ１（ｋ）～Ｒｎ（ｋ）、Ｃ１（ｋ）～Ｃｎ（ｋ）を行集合毎に用意し、局所メモリ１１ｂ～１ｎｂに領域を確保する。

【００３８】そして、要素プロセッサ１１ａ～１ｎａは、それぞれ第１～第ｎ行集合に含まれている値が「０」でない要素のエントリを、それぞれの局所メモリ１１ｂ～１ｎｂに領域を確保したエントリ集合Ｅ１～Ｅｎに追加していく。また、要素プロセッサ１１ａ～１ｎａは、エントリの追加時には、配列Ｒ１（ｋ）～Ｒｎ（ｋ）、Ｃ１（ｋ）～Ｃｎ（ｋ）の値を更新していく（ステップＳ２１１～Ｓ２１ｎ）。

【００３９】ｎ個のノード１１～１ｎにおいてステップＳ２１１～Ｓ２１ｎの処理が終了すると、並列コンピュータ１の１個のノード１１～１１のうちの１つのノード（ノード１１とする）の要素プロセッサ１１ａは、何番目の変数を消去するかを示す変数ｉを用意し、「１」を代入して初期化する。さらに、要素プロセッサ１１ａは、後述するステップＳ２４０の処理で交換された行及び列の数を示す置換配列Ｓ（ｋ）を用意し、局所メモリ１１ｂに領域を確保する。この時点において、置換配列Ｓ（ｋ）は行及び列が交換されていない状態で初期化される（ステップＳ２２０）。

【００４０】次に、要素プロセッサ１１ａは、ノード１１～１ｎの局所メモリ１１ｂ～１ｎｂに遠隔アクセスし、配列Ｒ１（ｋ）～Ｒｎ（ｋ）、Ｃ１（ｋ）～Ｃｎ（ｋ）を取得し、それぞれの要素を加算した配列Ｒ（ｋ）、Ｃ（ｋ）を求める。また、要素プロセッサ１１ａは、ノード１１～１ｎの局所メモリ１１ｂ～１ｎｂに遠隔アクセスし、エントリ集合Ｅ１～Ｅｎを取得し、その和集合であるエントリ集合Ｅを求める。

【００４１】次に、要素プロセッサ１１ａは、変数を消去するときに必要となる演算回数が最小となる変数を、まだ消去していない変数から求める。すなわち、ｋ番目の変数を消去するときに必要となる演算回数は、Ｒ（ｋ）×Ｃ（ｋ）によって求められるので、要素プロセッサ１１ａは、Ｒ（Ｓ（ｗ））×Ｃ（Ｓ（ｗ））が最小となるｗの値を求める。ここで、Ｒ（Ｓ（ｗ））、Ｃ（Ｓ（ｗ））の値を求めるときに、既に消去された変数と式に対応する行列部分のエントリや、対角要素となるエントリ数はカウントされない（ステップＳ２３０）。

【００４２】要素プロセッサ１１ａは、エントリ集合Ｅ

の第ｉ行と第ｗ行とを交換し、第ｉ列と第ｗ列とを交換する。この交換は、Ｓ（ｉ）の値とＳ（ｗ）の値とを交換することによって行われる（ステップＳ２４０）。

【００４３】要素プロセッサ１１ａは、次に、ステップＳ２４０の結果求められた置換配列Ｓ（ｋ）をメッセージとして相互結合網１０を介してノード１１～１ｎに送信する。置換配列Ｓ（ｋ）を受け取ったノード１１～１ｎは、再びそれぞれが並列に動作する。これらのノードの要素プロセッサ１１ａ～１ｎａはそれぞれ、第ｉ変数を消去するときに発生するフィルインを行集合毎に求め、そのフィルインに対応するエントリをエントリ集合Ｅ１～Ｅｎに追加する。

【００４４】ここで、右上三角行列の第ｉ行に含まれるすべてのエントリの位置が与えられれば、各行集合で発生するフィルインの位置は、行集合毎に並列に求めることができる。従って、フィルインに対応するエントリをエントリ集合Ｅ１～Ｅｎのそれぞれに追加する処理も並列に行うことが可能となる（ステップＳ２５１～Ｓ２５ｎ）。

【００４５】ノード１１～１ｎにおいて、ステップＳ２５１～Ｓ２５ｎの処理が終了すると、ノード１１の要素プロセッサ１１ａは、変数ｉの値を「１」だけインクリメントする（ステップＳ２６０）。要素プロセッサ１１ａは、さらに変数ｉの値がｍ－１の値以下であるかどうかを判定する（ステップＳ２７０）。

【００４６】ステップＳ２７０で変数ｉの値がｍ－１以下であると判定されているときは、ステップＳ２３０の処理に戻る。このような処理の繰り返しによって、最適なピボット順序とフィルインの発生位置が求められる。

【００４７】一方、ステップＳ２７０で変数ｉの値がｍの値に等しくなったと判定されると、要素プロセッサ１１ａは、ノード１１～１ｎの局所メモリ１１ｂ～１ｎｂに遠隔アクセスし、エントリ集合Ｅ１～Ｅｎを取得する。要素プロセッサ１１ａは、エントリ集合Ｅ１～Ｅｎを用いて行列を圧縮し、圧縮した行列データを局所メモリ１１ｂに格納する（ステップＳ２８０）。

【００４８】ここで、行列データの圧縮形式としては、例えば、"Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods"（R. Barrett他著、１９９４年刊）に記載されているような行圧縮形式を用いることができる。

【００４９】ステップＳ２８０の処理が終了すると、記号計算処理が終了し、次に、数値計算処理が実行される。そして、ワークステーション２から並列コンピュータ１に係数行列として渡された線形連立方程式の解が求められる。

【００５０】以下、この実施の形態における線形連立方程式の求解処理の具体例について、図４～図７を参照して説明する。ここで、ワークステーション２から並列コンピュータ１に渡される係数行列は、図４に示すものであり、図中の破線で示すように３つの行集合に分けられているものとする。

【００５１】各行集合を受け取ったノード（ノード１１～１３とする）では、ステップＳ２１１～Ｓ２１３において行集合毎に値が「０」でないエントリがエントリ集合Ｅ１～Ｅ３に追加される。このとき局所メモリ１１ｂ～１３ｂに記憶されるエントリ集合Ｅ１～Ｅ３は、図５（ａ）に示すように、

Ｅ１＝｛＜１，１＞，＜１，３＞，＜１，５＞，＜２，２＞，＜２，３＞，＜２，４＞，＜２，６＞｝

Ｅ２＝｛＜３，１＞，＜３，２＞，＜３，３＞，＜４，２＞，＜４，４＞，＜４，５＞，＜４，６＞｝

Ｅ３＝｛＜５，１＞，＜５，４＞，＜５，５＞，＜６，２＞，＜６，４＞，＜６，６＞｝

に設定される。

【００５２】また、局所メモリ１１ｂ～１３ｂに記憶されている、エントリ集合Ｅ１～Ｅ３へのエントリの追加時に更新されていく配列Ｒ１～Ｒ３、Ｃ１～Ｃ３は、図５（ａ）に示すように、

Ｒ１＝［２，３，０，０，０，０］

Ｃ１＝［０，０，２，１，１，１］

Ｒ２＝［０，０，２，３，０，０］

Ｃ２＝［１，２，０，０，１，１］

Ｒ３＝［０，０，０，０，２，２］

Ｃ３＝［１，１，０，２，０，０］

となる。

【００５３】そして、ステップＳ２２０において、ノード１１の局所メモリ１１ｂに記憶される変数ｉ及び置換配列Ｓが初期化され、図５（ａ）に示すように、

ｉ＝１

Ｓ＝［１，２，３，４，５，６］

に設定される。

【００５４】次に、ステップＳ２３０において、ピボット選択処理が行われる。ここで、

Ｒ＝［２，３，２，３，２，２］

Ｃ＝［２，３，２，３，２，２］

であるので、Ｒ（Ｓ（ｗ））×Ｃ（Ｓ（ｗ））を最小にするｗの値は、１となる。

【００５５】次に、ステップＳ２４０において、Ｓ（ｉ）とＳ（ｗ）とを入れ替える処理が行われるが、ｉ＝ｗであるので、実際には変化はない。もっとも、置換配列Ｓをノード１１～１３のすべてにメッセージとして送信する処理は行われる。

【００５６】次に、ステップＳ２５１～Ｓ２５３において、元の係数行列におけるＳ（ｉ）番目の変数を消去するときに生じるフィルインに対応するエントリが、ノード１１～１３のそれぞれでエントリ集合Ｅ１～Ｅ３に追加される。このとき、局所メモリ１１ｂ～１３ｂに記憶されるエントリ集合Ｅ１～Ｅ３は、図５（ｂ）に示すように、

E1＝{<1，1>，<1，3>，<1，5>，<2，2>，<2，3>，<2，4>，<2，6>}
E2＝{<3，1>，<3，2>，<3，3>，<3，5>，<4，2>，<4，4>，<4，5>，<4，6>}
E3＝{<5，1>，<5，3>，<5，4>，<5，5>，<6，2>，<6，4>，<6，6>}
に設定される。

【0057】また、配列R1～R3、C1～C3の値も更新され、図5（b）に示すように、
R1＝[0，3，0，0，0，0]
C1＝[0，0，1，1，0，1]
R2＝[0，0，2，3，0，0]
C2＝[1，2，0，0，1，1]
R3＝[0，0，0，0，2，2]
C3＝[0，1，1，2，0，0]
となる。

【0058】そして、ステップS260において変数iがインクリメントされ、図5（b）に示すように、i＝2となる。そして、再びステップS230の処理が実行される。

【0059】以上のような処理をi＝6－1＝5となるまで繰り返す。この過程において、ステップS251～S253では、図6に示すようにフィルインに対応するエントリがエントリ集合E1～E3に追加されていく。i＝5となったとき、エントリ集合E1～E3、配列R1～R3、C1～C3、変数i、及び置換配列Sは、図5（c）に示すように、次の通りとなる。

【0060】E1＝{<1，1>，<1，3>，<1，5>，<2，2>，<2，3>，<2，4>，<2，5>，<2，6>}
E2＝{<3，1>，<3，2>，<3，3>，<3，5>，<4，2>，<4，4>，<4，5>，<4，6>}
E3＝{<5，1>，<5，2>，<5，3>，<5，4>，<5，5>，<6，2>，<6，4>，<6，6>}

【0061】R1＝[0，0，0，0，0，0]
C1＝[0，0，0，0，0，0]
R2＝[0，0，0，0，0，0]
C2＝[0，0，0，0，0，0]
R3＝[0，0，0，0，0，0]
C3＝[0，0，0，0，0，0]
S＝[1，3，5，4，2，6]

【0062】そして、上記の処理で最終的に得られたエントリ集合E1～E3を用いて、ステップS280において、図7に示すように行列データが圧縮されて、ノード11の局所メモリ11bに格納される。

【0063】上記の処理でエントリ集合Eに追加されるエントリ数は、図5（c）に示すように、最終的には24個となっているが、行集合毎、すなわちノード毎に8個ずつに分散されている。従って、各ノード11～13におけるエントリ集合E1～E3のための局所メモリ11b～13bの使用量が、1つのノードで処理を行う場合に比べて3分の1で済むことがわかる。また、各ノード11～13は、係数行列全体を保持する必要がなく、その3分の1のデータ量で済む行集合を保持していればよい。従って、1つのノードにおける局所メモリの容量が従来のほぼ3分の1で済むことになり、また、局所メモリの容量が変わらないのであれば、従来よりもサイズの大きな係数行列（要素数がほぼ3倍）を扱えることとなる。

【0064】また、図6に示すように、ステップS251～S253のエントリの追加登録の処理は、ノード11～13で並列処理を行っていることにより、全体として9ステップで済んでいる。これに対して、1つのノードで同様の処理を行った場合には、追加登録の総数が17個であるので、17ステップを要することとなる。すなわち、この部分での計算速度がほぼ2倍になっていることがわかる。

【0065】以上説明したように、この実施の形態によれば、線形連立方程式の求解処理の最初の段階として実行される記号計算処理において、並列コンピュータ1の複数のノード11～1nが並列処理を行っている。

【0066】このとき、各ノード11～1nは、局所メモリ11a～1na内に係数行列を分割した行集合のみを保持すればよく、局所メモリ11a～1naに記憶されるまた、エントリ集合E1～E3も行集合に対応するもののみで足りることとなる。このため、同一の係数行列に対して計算を行うならば、局所メモリ11b～1ncの容量が従来例よりも少なくて済む。また、従来例と局所メモリ11b～1ncの容量が同一であるならば、従来例よりも大きなサイズの係数行列を扱うことが可能となる。

【0067】また、エントリ集合E1～E3へのエントリへの追加を、複数のノードで分散して行うことができる。このため、従来例に比べてエントリの追加に要する計算時間が短くて済むようになる。

【0068】また、ノード11から各ノード11～1nへ置換配列Sを送ることによって、ノード間で行列或いは行集合のやり取りをしなくても、各ノード11～1nでフィルインを求めることが可能となる。

【0069】［第2の実施の形態］この実施の形態では、線形連立方程式の解を求めることによる並列回路のシミュレーションに、本発明を適用した場合について説明する。

【0070】この実施の形態においても、並列回路シミュレーションを行うハードウェアとして、図1に示す並列コンピュータ1を用いることができる。但し、この実施の形態において、シミュレーション対象となる並列回

路の回路接続情報は、ワークステーション２の外部記憶装置２２にライブラリとして登録されており、入力装置２１からの指示に従って、ライブラリから読み出されて並列コンピュータ１に投入される。

【００７１】以下、この実施の形態において、図１の並列コンピュータ１で実行される並列回路シミュレーションの処理を、図９のフローチャートを参照して説明する。

【００７２】処理が開始すると、まず、ユーザは入力装置２１を操作して、外部記憶装置２２にライブラリとして登録されている回路接続情報をワークステーション２から相互結合網１０を介して並列コンピュータ１に入力する（ステップＳ３００）。

【００７３】並列コンピュータ１のノード１１～１１の要素プロセッサ１１ａ～１１ａのうち、回路接続情報を受け取ったノード（ノード１１とする）の要素プロセッサ１１ａは、入力された回路接続情報によって示される回路を、ノード１１～１１の数と同数の１個の部分回路に分割する。この部分回路の分割は、例えば、特開平９－３１９７８４号公報に記載されている方法によって行う。そして、ノード１１の要素プロセッサ１１ａは、分割した１個の部分回路に関する情報をそれぞれ、ノード１１～１１に相互結合網１０を介し（但し、ノード１１には、プロセス間通信によって）て送信する（ステップＳ３１０）。

【００７４】部分回路に関する情報を受け取ったノード１１～１１は、それぞれが並列に動作する。これらのノード１１～１１の要素プロセッサ１１ａ～１１ａは、それぞれ部分回路に関する情報を、後述するシミュレーション時に効率がよい形式のオブジェクトデータ（部分回路を示す行列と部分回路内の各素子のパラメータ）に変換するコンパイル作業を行う（ステップＳ３２１～Ｓ３２１）。

【００７５】更に、要素プロセッサ１１ａ～１１ａは、各部分回路毎に線形連立方程式の求解のための記号処理を行う。この線形連立方程式の求解のための記号処理は、図９のフローチャートに従って行われる（ステップＳ３３１～Ｓ３３１）。更に、要素プロセッサ１１ａ～１１ａは、それぞれステップＳ３３１～Ｓ３３１で記号処理した結果に対して、数値計算処理を行い、その結果を１つのノード（ここでは、ノード１１とする）に相互結合網１０を介して（但し、ノード１１ではプロセス間通信によって）送信する（ステップＳ３４１～Ｓ３４１）。

【００７６】各部分回路の演算結果を受信したノード１１の要素プロセッサ１１ａは、各部分回路の縮退モデルのオブジェクトデータと結合回路情報とを用いて、結合回路のコンパイルを行い、結合回路を示す行列を生成する（ステップＳ３５０）。

【００７７】次に、要素プロセッサ１１ａは、生成した行列を複数の行列集合に分ける。そして、図３のフローチャートに示す処理により、複数のノードが並列動作して結合回路方程式の求解のための記号計算処理が行われる（ステップＳ３６０）。

【００７８】ステップＳ３６０の処理で記号計算処理がされ、行列が圧縮格納されると、その行列を格納している局所メモリ（ここでは、局所メモリ１１ｂとする）を有するノード１１の要素プロセッサ１１ａは、図２のフローチャートに示すステップＳ１４０及びＳ１５０の処理を行って、数値計算処理を行う（ステップＳ３７０）。ノード１１の要素プロセッサ１１ａは、得られた結果を波形として相互結合網１０を介してワークステーション２に出力する（ステップＳ３８０）。そして、このフローチャートの処理を終了する。

【００７９】以上説明したように、この実施の形態では、結合回路方程式の求解を行う場合に、複数のノード１１～１１での並列処理が可能となる。このため、回路シミュレーションを従来よりも高速に行うことができ、また、局所メモリ１１ｂ～１１ｃの容量による制限も受けなくなるため、大規模な回路のシミュレーションを行うことができる。

【００８０】［実施の形態の変形］本発明は、上記の第１、第２の実施の形態に限られず、様々な変形が可能である。以下、上記の実施の形態の変形態様について、説明する。

【００８１】上記の第１の実施の形態では、ワークステーション２から処理を依頼された係数行列は予めｎ個の行集合に分割されているものとして説明した。しかしながら、ワークステーション２から並列コンピュータ１に処理を依頼する係数行列は、行集合に分割されていないものであってもよい。この場合、処理依頼を受け取ったノードが、並列コンピュータ１のノード１１～１１の数ｋに応じて係数行列を行集合に分割すればよい。

【００８２】上記の第１、第２の実施の形態では、処理対象となるデータ（線形連立方程式の係数行列或いは回路接続情報）の投入以外のすべての処理を図１に示す並列コンピュータ１で行っていた。しかしながら、例えば、記号計算処理を並列コンピュータ１で行い、数値計算処理を並列コンピュータ１とは別のベクトル計算機で行うことも可能である。また、並列処理される部分以外をワークステーション２で実行し、並列処理される部分の処理をワークステーション２０から並列コンピュータ１に依頼することも可能である。

【００８３】上記の第２の実施の形態では、部分回路の線形連立方程式を求解するための記号計算処理は、部分回路毎に１つのノードで行われていた。しかしながら、この処理も、図３のフローチャートで示したのと同様にして複数のノードで並列に実行するものとしてもよい。

【００８４】

【発明の効果】以上説明したように、本発明によれば、

線形連立方程式の解を求める場合の記号計算において、計算時間を短くしたり、扱える行列の規模を大きくしたりすることができる。

【００８５】また、大規模な回路のシミュレーションを高速に行うことができる。

【図面の簡単な説明】

【図１】本発明の第１の実施の形態に適用される並列コンピュータの構成を示すブロック図である。

【図２】本発明の第１の実施の形態にかかる線形連立方程式の求解処理の概略を示すフローチャートである。

【図３】図２の記号計算処理を詳細に示すフローチャートである。

【図４】本発明の第１の実施の形態における具体例で、ワークステーションから並列コンピュータに投入される係数行列を示す図である。

【図５】（ａ）～（ｃ）は、本発明の第１の実施の形態における具体例での記号計算処理を説明する図である。

【図６】本発明の第１の実施の形態における具体例でのエントリが追加登録される順序を示す図である。

【図７】本発明の第１の実施の形態における具体例での行列の圧縮格納用データ構造を示す図である。

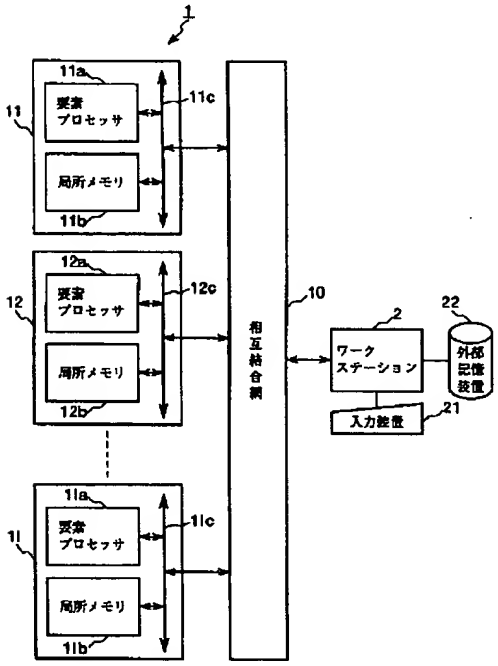
【図８】本発明の第２の実施の形態にかかる並列回路シミュレーションの処理を示すフローチャートである。

【図９】従来にかかる記号計算処理を示すフローチャートである。

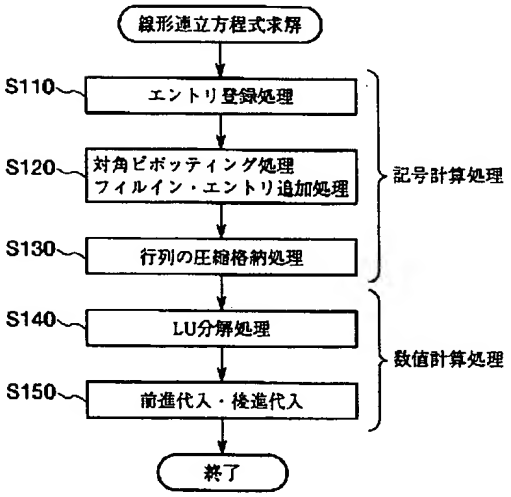
【符号の説明】

１　　並列コンピュータ
１０　相互結合網
１１～１１　ノード
１１ａ～１１ａ　要素プロセッサ
１１ｂ～１１ｂ　局所メモリ
１１ｃ～１１ｃ　バス
２　　ワークステーション
２１　入力装置
２２　外部記憶装置

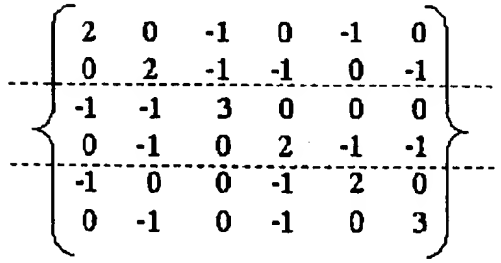
【図１】



【図２】

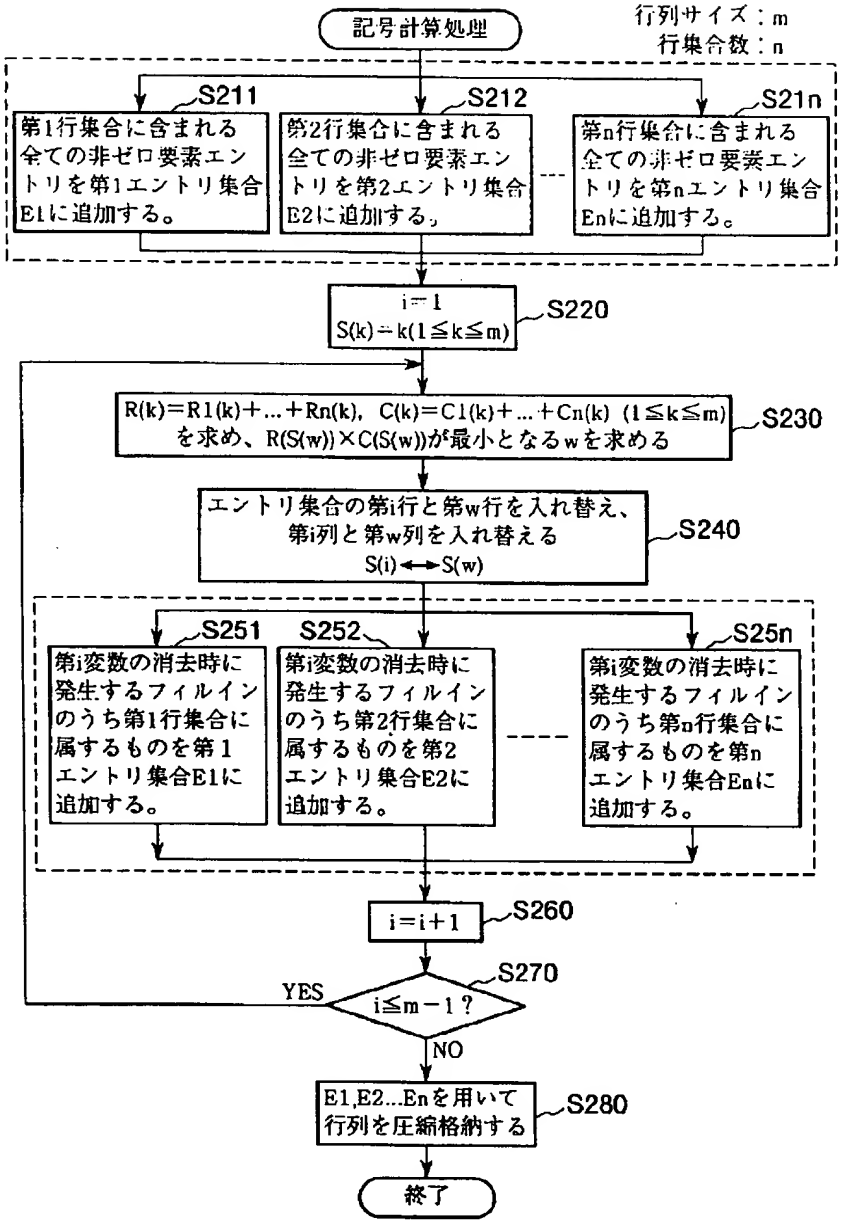


【図４】

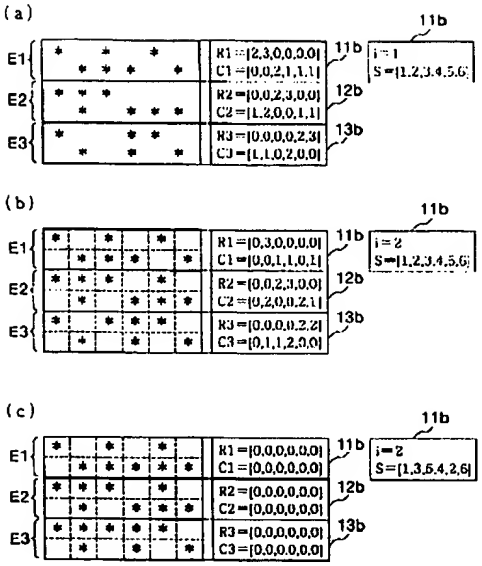


2	0	-1	0	-1	0
0	2	-1	-1	0	-1
-1	-1	3	0	0	0
0	-1	0	2	-1	-1
-1	0	0	-1	2	0
0	-1	0	-1	0	3

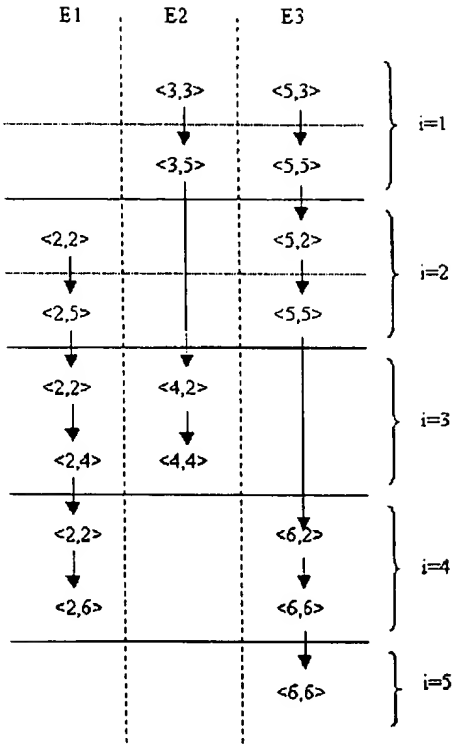
【図3】


記号計算処理
行列サイズ：m
行集合数：n
S211
第1行集合に含まれる全ての非ゼロ要素エントリを第1エントリ集合E1に追加する。
S212
第2行集合に含まれる全ての非ゼロ要素エントリを第2エントリ集合E2に追加する。
S21n
第n行集合に含まれる全ての非ゼロ要素エントリを第nエントリ集合Enに追加する。
i=1
S(k)=k(1≦k≦m)
S220
R(k)=R1(k)+...+Rn(k), C(k)=C1(k)+...+Cn(k) (1≦k≦m)
を求め、R(S(w))×C(S(w))が最小となるwを求める
S230
エントリ集合の第i行と第w行を入れ替え、
第i列と第w列を入れ替える
S(i)←→S(w)
S240
S251
第i変数の消去時に発生するフィルインのうち第1行集合に属するものを第1エントリ集合E1に追加する。
S252
第i変数の消去時に発生するフィルインのうち第2行集合に属するものを第2エントリ集合E2に追加する。
S25n
第i変数の消去時に発生するフィルインのうち第n行集合に属するものを第nエントリ集合Enに追加する。
i=i+1
S260
YES
i≦m−1？
S270
NO
E1,E2...Enを用いて行列を圧縮格納する
S280
終了

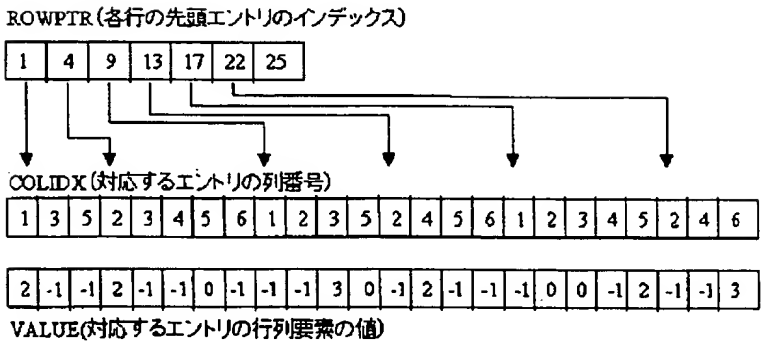
【図５】


(a)
E1
E2
E3
R1=[2,3,0,0,0,0]
C1=[0,0,2,1,1,1]
R2=[0,0,2,3,0,0]
C2=[1,2,0,0,1,1]
R3=[0,0,0,0,2,3]
C3=[1,1,0,2,0,0]
11b
12b
13b
i=1
S=[1,2,3,4,5,6]
(b)
R1=[0,3,0,0,0,0]
C1=[0,0,1,1,0,1]
R2=[0,0,2,3,0,0]
C2=[0,2,0,0,2,1]
R3=[0,0,0,0,2,2]
C3=[0,1,1,2,0,0]
i=2
S=[1,2,3,4,5,6]
(c)
R1=[0,0,0,0,0,0]
C1=[0,0,0,0,0,0]
R2=[0,0,0,0,0,0]
C2=[0,0,0,0,0,0]
R3=[0,0,0,0,0,0]
C3=[0,0,0,0,0,0]
i=2
S=[1,3,5,4,2,6]

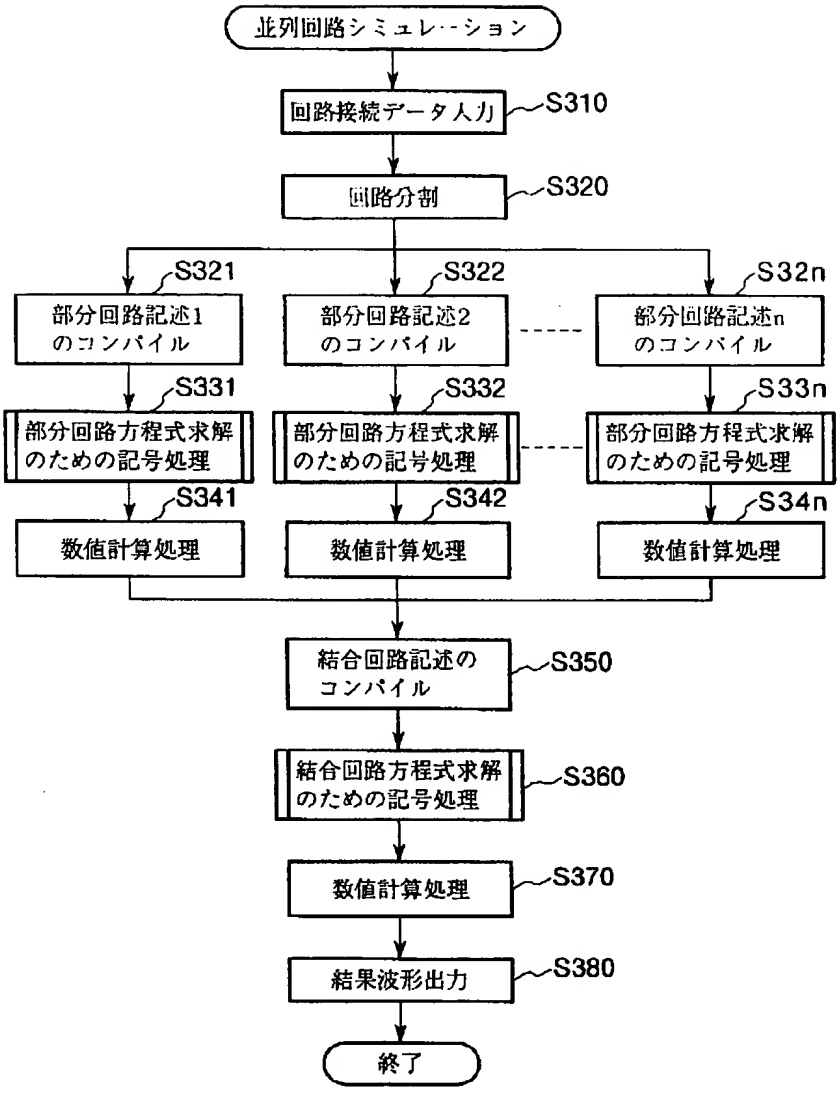
【図６】


E1
E2
E3
<3,3>
<5,3>
<3,5>
<5,5>
i=1
<2,2>
<5,2>
<2,5>
<5,5>
i=2
<2,2>
<4,2>
<2,4>
<4,4>
i=3
<2,2>
<6,2>
<2,6>
<6,6>
i=4
<6,6>
i=5

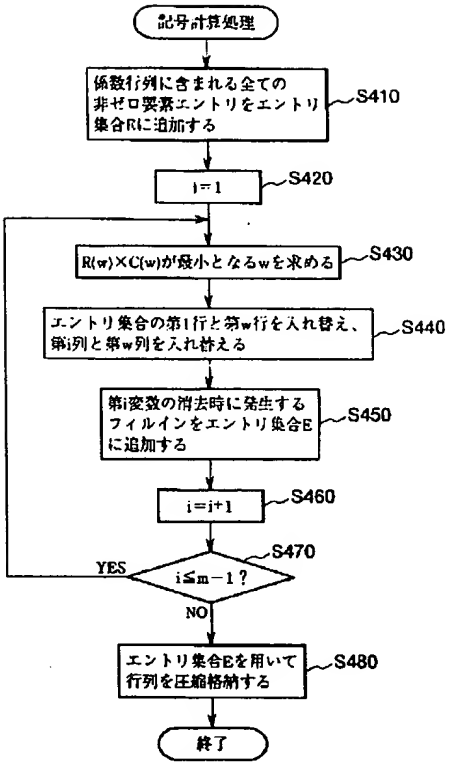
【図７】


ROWPTR(各行の先頭エントリのインデックス)
1 4 9 13 17 22 25
COLIDX(対応するエントリの列番号)
1 3 5 2 3 4 5 6 1 2 3 5 2 4 5 6 1 2 3 4 5 2 4 6
2 -1 -1 2 -1 -1 0 -1 -1 -1 3 0 -1 2 -1 -1 -1 0 0 -1 2 -1 -1 3
VALUE(対応するエントリの行列要素の値)

【図8】


並列回路シミュレーション
回路接続データ入力
S310
回路分割
S320
S321
S322
S32n
部分回路記述1のコンパイル
部分回路記述2のコンパイル
部分回路記述nのコンパイル
S331
S332
S33n
部分回路方程式求解のための記号処理
部分回路方程式求解のための記号処理
部分回路方程式求解のための記号処理
S341
S342
S34n
数値計算処理
数値計算処理
数値計算処理
結合回路記述のコンパイル
S350
結合回路方程式求解のための記号処理
S360
数値計算処理
S370
結果波形出力
S380
終了

【図９】


記号計算処理
係数行列に含まれる全ての非ゼロ要素エントリをエントリ集合Rに追加する
S410
i=1
S420
R(w)×C(w)が最小となるwを求める
S430
エントリ集合の第i行と第w行を入れ替え、第i列と第w列を入れ替える
S440
第i変数の消去時に発生するフィルインをエントリ集合Eに追加する
S450
i=i+1
S460
YES
i≦m−1？
S470
NO
エントリ集合Eを用いて行列を圧縮格納する
S480
終了